Towards a tool-supported approach for collaborative process modeling and enactment

Komlan Akpédjé Kedji, Minh Tu Ton That, Bernard Coulette, Redouane Lbath, Hanh Nhi Tran, Mahmoud Nassar

Towards a tool-supported approach for collaborative process modeling and enactment

Komlan Akpédjé Kedji[1,3], Minh Tu Ton That[1], Bernard Coulette[1], Redouane Lbath[1], Hanh Nhi Tran[2], Mahmoud Nassar[3]

[1]IRIT, Macao team, Toulouse, France
{kedji,coulette,lbath}@univ-tlse2.fr, minhtutonthat@gmail.com

[2] LISyC ENSTA-Bretagne, Université Européenne de Bretagne, Brest, France
hanh_nhi.tran@ensta-bretagne.fr

[3] ENSIAS, SIME, IMS team, Mohamed V Souissi University, Rabat, Morocco
nassar@ensias.ma

***Abstract*— In software engineering, as in any collective endeavor, understanding and supporting collaboration is a major concern. Unfortunately, the main concepts of popular process formalisms are not always adequate to describe collaboration. We extend the Software & System Process Engineering Meta-Model (SPEM) by introducing concepts needed to represent precise and dynamic collaboration setups that practitioners create to address ever-changing challenges. Our goal is to give practitioners the ability to express evolving understanding about collaboration in a formalism suited for easy representation and tool-provided assistance. Our work is based on a collaborative process metamodel we have developed. In this paper, we first present a meta-process for process modeling and enactment, which we apply to our collaborative process metamodel. Then we describe the implementation of a suitable process model editor, and a project plan generator from process models.**

Keywords*— *Collaborative processes; CM_SPEM metamodel; Generation of project plans; ATL transformation language

I. INTRODUCTION

Collaboration can be simply defined as the act of working together, towards a common goal. It is thus a pervasive concern in any collective endeavor.

Trying to support collaboration using process formalisms and standards like SPEM highlights how inadequate they are for the purpose of process instantiation. For example, SPEM considers issues like the number of people affected to a task, and what each one is doing, as enactment concerns, which are not to be described in the process model. However, it is not at all clear how a process model is to be used in a project; given that it has no concept to represent project-level entities (like the actual people playing roles). In the case of collaboration, we argue that, for example, real people do structure collaboration, and should therefore be represented in process models, if collaboration support is envisioned. Our extension to SPEM thus aims to be a better mental model, and enable a more accurate description of project-level issues.

Process-related tools usually consider the process model as a somewhat rigid specification, and strive to make sure rules defined in the model are followed. We consider process models as living descriptions of decisions about the realization of the project. This description is then used to automate, whenever possible and worthwhile, day to day decisions made by practitioners and represented in the process model. This naturally results in a feedback loop, where the process models guides short-term actions, and those actions in turn help improve the process model.

As taking project-level issues into account in process models tends to blur the line between modeling and enactment, we propose a meta-process for process modeling and enactment. This meta-process highlights two ways of going from process modeling to process enactment: using a PSEE (Process centered Software Engineering Environment) or using an off-the-shelf project management tool. In this paper, we present the second alternative. To assist process designers and project managers during the meta-process, we developed an editor for collaborative process models, and a project plan generator targeting two popular project planning tools (MS Project [1] and Gantt Project [2]). Our ultimate goal is to support modeling and enacting collaborative processes. However, in this report on a work in progress, our main contribution relates to process enactment.

This work has been done in the context of the Galaxy Project, funded by ANR, France. Galaxy addresses the collaborative development of complex systems using highly heterogeneous development environments and following the model driven engineering approach. Our role in Galaxy is focused on collaborative processes modeling and enactment. Our CM_SPEM metamodel is one of the Galaxy project's deliverables. It is being validated via an industrial case study provided by Airbus. In this paper, we present our approach for collaborative process modeling, and describe two supporting tools which have been developed in the context of the Galaxy project.

The rest of this paper is structured as follows. Section II describes our general meta-process for process modeling and enactment. Section III presents the metamodel CM_SPEM, and how it can be used to describe a sample collaboration situation. The development of the process editor and of the project plan generator is described in section IV. Section V discusses some related works and section VI concludes the paper.

II. A GENERIC META-PROCESS FOR PROCESS MODELING AND ENACTMENT

Many approaches address the way one can model and enact software processes, but few of them describe the process elaboration itself. In [3], the authors stipulate the life cycle of a development process which consists of four stages: defining the process meta-model, building the process model, applying the process model in the project and improving the process. Similarly, in [2], three stages are proposed for developing the process model, tailoring the structural project plan and generating the project plan. Having taken these approaches into consideration, we have defined a general meta-process. Figure 1 shows this meta-process in SPEM format. It consists of 4 stages which are described in the following sub-sections.

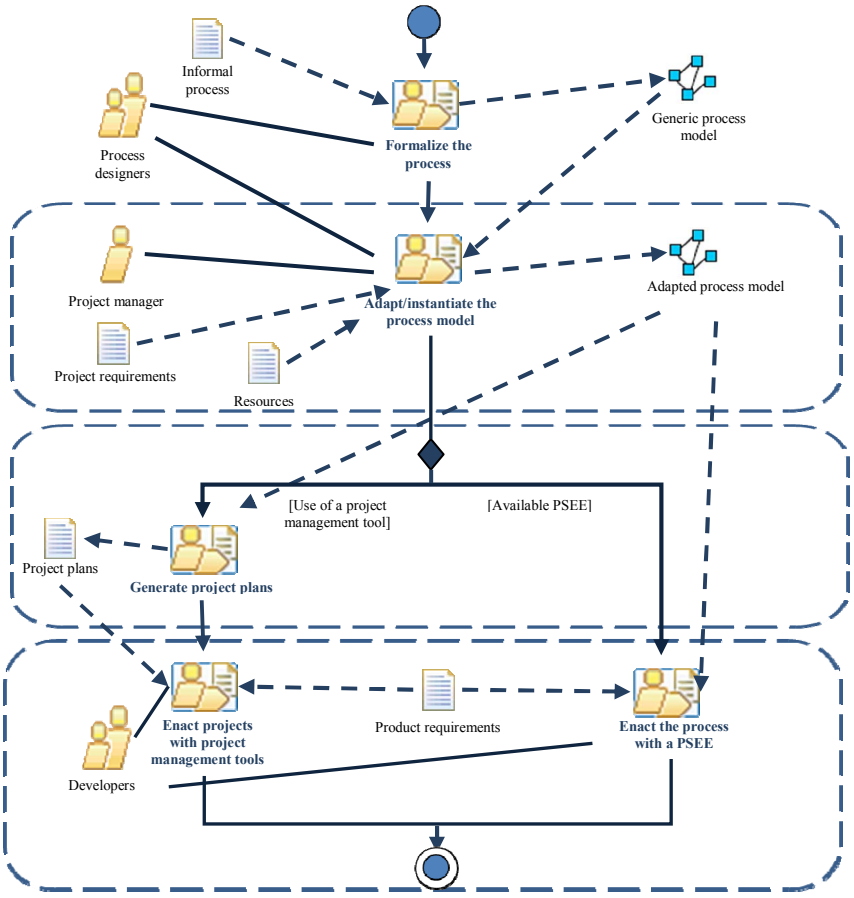


Figure 1. Meta-process for modeling and enactment of processes (using the SPEM graphical notation)

A. *Stage 1: Formalize the process*

This stage is performed by actors playing the "process designer" role. To formalize and model a process, we have to consider two scenarios. A common scenario consists in defining a new process from scratch. In this case the input product of this phase is generally an informal process resulting from an analysis of existing practices including interviews of project managers, etc. It may be described as textual documentation. The second scenario occurs when the process to formalize is an extension/adaptation of an existing one, that may be already formalized or not. Such a defined and therefore repeatable process offers the chance to incorporate the knowledge and the lessons learned of many seasoned project managers into active projects [4]. In this case, one may apply reuse techniques (based on mappings) as those existing in SPEM. The phase consists in producing a generic process model in both scenarios, independent of any specific project - described in a given PML (Process Modeling Language).

B. *Stage 2: Adapt/Instantiate a process model*

This stage is performed whenever one wants to adapt the generic process model previously defined (in the first

stage) to a given project. It is performed by the project manager and at least one process designer who should collaborate with him/her during this phase. This stage mostly consists in instantiating the elements of the generic process model and allocating resources to them. Those resources are people, tools available at project execution time. For instance, a task can be instantiated with a starting date and a duration, actors (human developers) playing a given role can be allocated to a given task in conformity to the generic process model; a tool can be allocated to an automatic transformation, etc. In addition, the process designer and the project manager may adapt/refine the generic process model to take into account the project's characteristics. The result of this stage is a formalized process model called "enactable process model". This process model is sufficiently refined to be exploited in the next stage.

C. Stage 3: Generate the project plan

This stage takes place whenever we intend to use an existing off-the-shelf project management tool at enactment time. It is usually the case when no PSEE is available. Such stage may be performed automatically by a transformation engine to produce a project plan from an adapted process model. The project plan plays an important role in the process development. It provides the foundation for measuring project progress, processes, and products [8]. Using the project plan like an intermediate product before moving to the process enactment stage helps benefit from the functionalities supported by project management tools

D. Stage 4: Perform a project

This stage takes place at enactment time. It is performed by developers who work together to run a given project development. The input parameters are product requirements of the project and the adapted process model. The nature of these requirements (e.g.: users' requirements, analysis model, design model, etc.) depends on the project's type. The result of this stage is a product (e.g.: executable code, documentation, model, etc.). This stage is supported by a project management tool or by a PSEE. During this stage, if the supporting environment is a true PSEE, developers are guided and assisted in performing their tasks according to the adapted process model. In case of a project management tool, the project plan is a key reference source to govern the project execution. The project manager and maybe team managers may dynamically make the current process model evolve.

III. COLLABORATIVE PROCESSES MODELING WITH CM_SPEM

A. Introduction

Our approach is based on the insight that in an ongoing software engineering project, where issues like collaboration are manifest, the concepts involved for the practitioners are the actual people doing the job, what each of them is doing, and which artifacts they are manipulating. However, process models are usually described using roles (a role may be played by different people, and someone may play different roles), products (a product such as a source file may have different physical representations in different workspaces), and tasks (a task may be carried out by different people, each focusing on a specific part of a product). We introduce concepts to account for these precisions, and describe how they relate to each other.

B. The CM_SPEM metamodel

Our conceptualization is presented as a metamodel, CM_SPEM (Collaborative Model-based Software & Systems Process Engineering Metamodel), that extends SPEM [5] with ad-hoc collaboration description capabilities. A more detailed description of the metamodel – including semantic aspects - is available in [11] and [12].

In addition to the static concepts described hereafter, CM_SPEM has an event-based dynamic semantic [12], which allows models based on it to evolve over time, in response to the availability of new information (modeled as events). Similarly, any modification to a CM_SPEM process model (like the addition of a new participant) generates an event. This allows third party tools to listen to specific events on the process model, and offer some assistance to participants when appropriate (for example, a standard development environment can be set up for a new participant, with all necessary documentation).

1) Central Concepts

The three new central concepts introduced in CM_SPEM are:

- *Actor*: it unambiguously identifies a specific human participant in a project.
- *Actor Specific Task*: it is a unit of work done by a specific actor, towards the execution of a *TaskUse*
- *Actor Specific Artifact* is a physical occurrence of a *Work Product Use*, in the personal workspace of a specific actor. This is the personal copy of the actor, and is manipulated only by him/her.

Figure 2 is a succinct representation of the metamodel which focuses on collaborative aspects. The three central concepts are linked to the SPEM concepts (shaded) they add precision to.

2) Relationships

Relationships encode knowledge about collaboration, and come in two groups.

Relationships in the first group are used to relate the central concepts one to another: *Task Assignment* (relates an *Actor* to an *Actor Specific Task* assigned to him/her), *Artifact Ownership* (relates an *Actor* to an *Actor Specific Artifact* that belongs to his/her workspace), and *Artifact Use* (relates an *Actor Specific Task* to an *Actor Specific*

Artifact that is manipulated when carrying the task out).

To describe the interactions of a couple of instances of one of the central concepts, the second group of relationships is used: *Actor Relationship*, *Actor Specific Task Relationship*, and *Actor Specific Artifact Relationship*. All relationships can be refined with the SPEM *Kind* mechanism, so that user-defined qualifications can be applied to them, to describe the precise knowledge about collaboration that they embody.

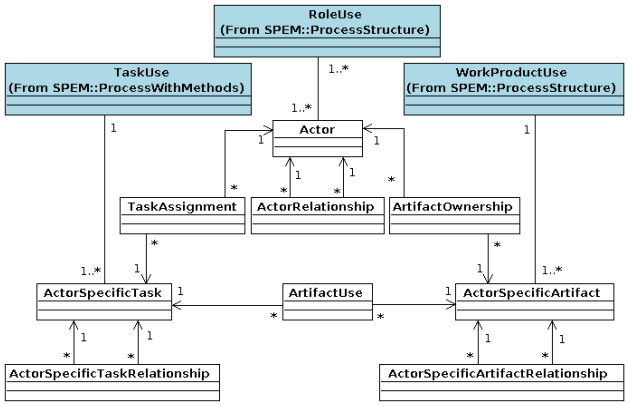


Figure 2. The CM_SPEM metamodel (extract)

C. *Example of a model conforming to CM_SPEM*

We consider a software engineering project concerned with the development of a complex ticket reservation system. The team is composed of the following participants:

- Bob (designer) designs and writes architecture description models, used to generate interfaces and data conversion code.
- Alice (integration manager) decides when features and fixes are ready for production, and merges them while making sure the result is functional and reasonably bug-free.
- Fred (deployment manager) deploys the project to production, monitors execution, and reports errors back to developers.
- Karl and Mike (developers) write code which implements performance sensitive functionalities and integration with legacy systems.
- Tracy (developer) writes integration tests for the system under development.

Figure 3 shows how the various actors and their relationships can be modeled in CM_SPEM (the role associated with each actor is written in square brackets).

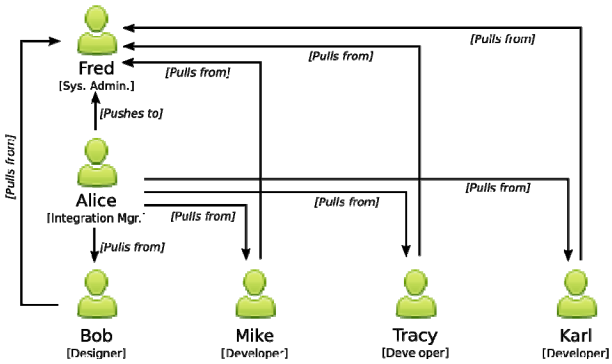


Figure 3. Actors and theirs relationships as part of a process model conforming to CM_SEPM

The relationships (represented with arrows) describe the general flow of work. Each feature is first prototyped in a developer's local repository. The integration manager (Alice) then "Pulls (contributions) from" developer repositories, integrates them, and then "Pushes (the result) to" production. Each developer "Pulls (artifacts) from" the official repository to bring his/her local repositories up to date. A detailed illustration of CM_SPEM is available in [12].

IV. PROCESS MODEL EDITOR AND PROJECT PLAN GENERATOR

On the one hand, we developed a graphical editor to support the creation of CM_SPEM models. On the other hand we developed a generator taking an adapted process model as input, and producing a project plan as output. Both of these tools were elaborated through MDE techniques.

A. *Development of the CM_SPEM editor*

CM_SPEM Editor is a tool for the design of process models. In the meta-process proposed in the section II, our editor serves at the first and the second stage. In other words, it supports the design of a generic process model and its adaptation to a specific project. When using the MDE (Model Driven Engineering) approach, the main purpose of our editor is assuring a stable development of process models that always conform to the defined CM_SPEM meta-model. To achieve this, we chose the TOPCASED [9] environment which is dedicated to MDE, and create, on the one hand, a tree-structured editor and, on the other hand, a graphical editor supporting the design of CM_SPEM models. Figure 4 shows the editor, with a model representing actors and their relationships, as described in the example of Section III (Subsection C).

CM_SPEM Editor makes a contribution to the conception of collaborative process models. It supports 7 types of diagrams: *Process Model Diagram f*or the design of generic process models, *ActorRelationship Diagram* for the specification of actors and their relationships,

ActorSpecificTaskRelationship Diagram for the specification of tasks and their relationships, *ActorSpecificArtifactRelationship Diagram* for the specification of artifacts and their relationships, *Kind Diagram* for the specification of relationship types and finally *CM_SPEM Diagram* for the assignment of tasks and resources. Besides designing functionalities supported, a remarkable feature of this generated editor is the model verification tool. This capacity assures that we always obtain models well conforming to their meta-models.

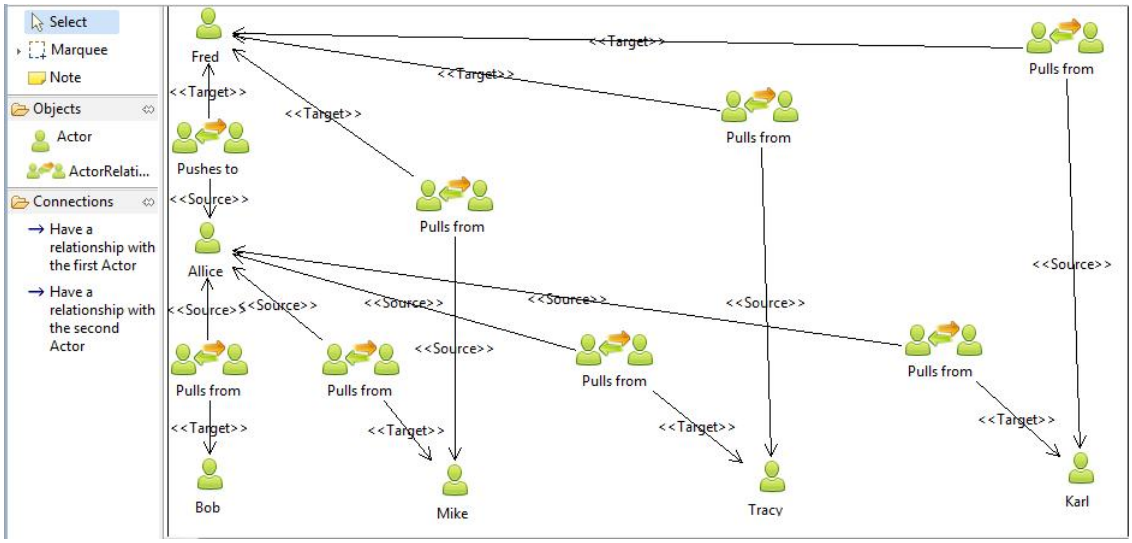


Figure 4. An *ActorRelationship Diagram* created in using the CM_SPEM Editor

B. Generation of project plans

The project plan generator is a means of producing project plans from process models which are designed by CM_SPEM editor. It aims at the third stage in the meta-process proposed in the section II. We conducted a survey of 13 existing management tools (using criteria like task, resource and actor description support). Despite the lack of collaborative support in these tools, we argue they still play an important role in the process development. Thank to their important functionalities (measuring project progress, processes, and products [8]), project management tools can be used as intermediary supports before moving to the process enactment stage in which PSEEs can fulfill collaborative requirements. Among the surveyed tools, we chose Gantt Project [2] and MSProject 2010 [1] to apply the transformation because they are popular and fit most of our criteria [21].

We opted for ATL to perform our transformations thanks to its simple rule architecture and its support of OCL (Object Constraint Language).

1) Transformations principle

The transformations from process models to project plans are PIM (Platform Independent Models) to PSM (Platform Specific Models) transformations in the MDA (Model Driven Architecture) approach. While process models are independent from any technical foundation, project plans depend critically on the tools by which they are designed. For instance, an abstract process model contains enough information to describe 2 common types of resource in a project: human resource and material resource. In a Microsoft Project 2010 project plan, these information can be considered as "Work" and "Material", while in a Gantt Project project plan, they are not pre-defined. A great care must be therefore taken during the creation of transformation rules to make sure the proper formats of project plans are created.

To be more specific, we outline in Listing 1 some important rules transforming the CM_SPEM elements into MSProject elements.

```
rule Task2Task   --transform a task to a task
from     s : CM_SPEM!ActorSpecificTask
to       t : MSProject!Task

rule Actor2Resource -- transform an actor to a resource
from     s : CM_SPEM!Actor
to       t : MSProject!Resource

rule Artifact2Resource --transform an artifact to a
resource
from     s : CM_SPEM!ActorSpecificArtifact
to       t : MSProject!Resource

rule  TaskAssignment2Allocation  --  transform  a  task
assignment to an allocation
from     s : CM_SPEM!TaskAssignment
to       t : MSProject!Allocation

rule ArtifactUse2Allocation -- transform an artifact
use to an allocation
from     s : CM_SPEM!ArtifactUse
to       t : MSProject!Allocation
```

Listing 1. CM_SPEM 2MSProject transformation rules

a) Models representing CIM level conform to CM_SPEM meta-model which is described in Section III.

b) Models representing PSM level conform to meta-models representing different specific project management tools. In case of Microsoft Project 2010, we defined the MSProject meta-model which captures basic charateristics of a MS Project 2010 project plan (Figure 5). For instance, the three most important concepts of a project plan – task, actor, artifact – are represented through 2 classes: Task and Resource. A task is specified with a name, a beginning day, a duration. A resource is given a name and distinguished by its type, etc.

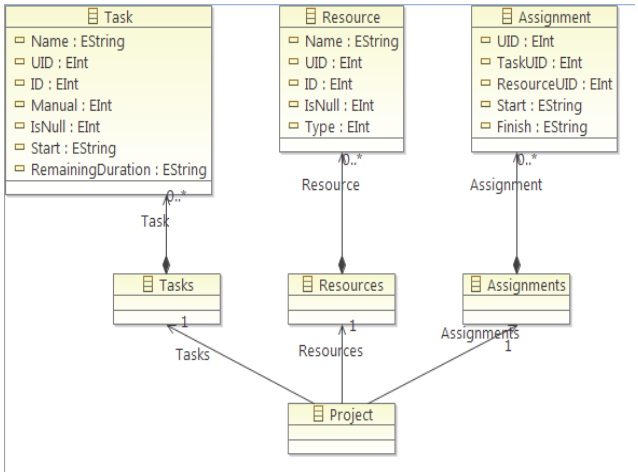


Figure 5. The MS Project Metamodel

2) Transformation from CM_SPEM to MS Project

Let us take the example of the transformation of CM_SPEM models to Microsoft Project 2010 format. Figure 6 represents the process of transformation in the MDE three-level view. The purpose of this transformation is to generate a XML document conforming to the Project XML Schema [10] from a process model conforming to CM_SPEM meta-model.

To achieve this purpose, we use 2 steps of transformation. The first one is from CM_SPEM model to MSProject model. The second one is from MSProject model to XML document. While the first transformation rule is written in the format of ATL "helpers" – a means of generating models conforming to pre-defined ECORE meta-models, the second transformation rule is written in the format of ATL "queries" – a means of extracting information from a model to a text. What we achieve finally is a project plan that can be used as input by Microsoft Project 2010.

Figure 7 is the CM_SPEM model of the above collaborative scenario designed by our editor and Figure 8 is a Microsoft Project 2010 project plan in form of a Gantt Project generated from the aforementioned model. In this project plan, the task "Write tests" is implemented collaboratively by two actors Alice and Tracy while using two artifacts Test1 and Test2.

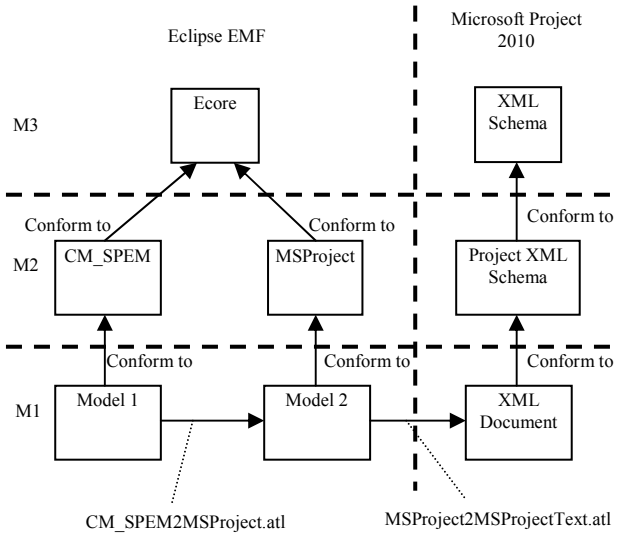
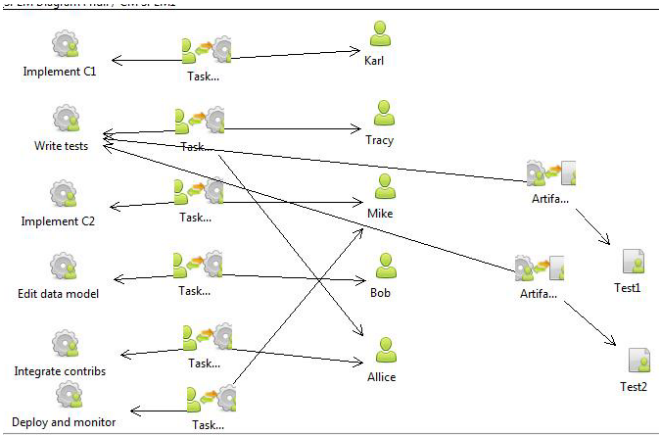


Figure 6. Transformation of a CM_SPEM model into a Project plan for Microsoft Project



. Figure 7. CM_SPEM Model

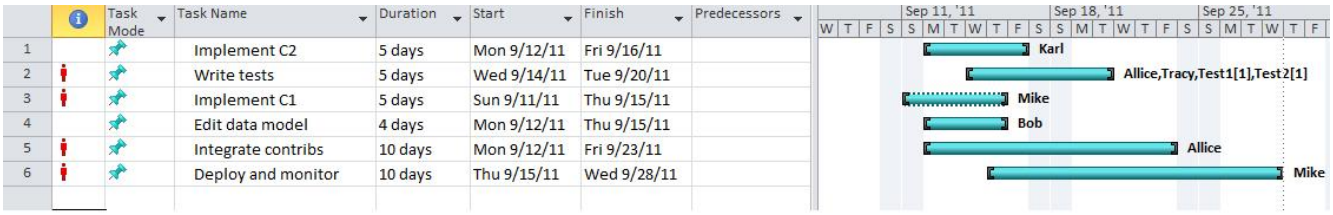


Figure 8. Generated Microsoft Project Project plan

I. RELATED WORKS

Previous contributions of interest to our work relate to the difficulty of formalizing knowledge about work practices, more flexible instantiation of process models, extending process metamodels like SPEM to allow a more precise definition of process models, and giving practitioners the ability to enhance process models with their evolving understanding of what they are doing

In [13], M. Polanyi et al., discuss how practitioners rely on implicit knowledge that is really hard if not impossible to formalize, and sums it up as "we know more than we can tell". It is even harder to formalize knowledge in the abstract, outside of any practical situation. Our approach builds upon this fundamental realization, by giving practitioners the ability to lightly represent their ongoing interactions, thus departing from the approaches that seek to embed ready-made knowledge once for all in process models.

Killisperger et al., developed a framework for flexible process instantiation [14].The goal is to assist step-by-step tailoring and instantiation of generic and complex process models. While the progressive approach is shared by our proposal, the goal of the framework in [14] is to make sure tailoring and instantiation respect certain pre-defined rules (syntax and organizational rules), while we focus on supporting collaboration.

There have been other efforts to extend SPEM. For example, [15] defines a formalism based on petri nets geared towards precise definition of MDE (Model Driven engineering) process models and process execution tracking. It however ignores resource allocation and roles definition, and concentrates on process steps (described with links, models, flow, resource, etc.) This formalism is more detailed than SPEM, but follows the same "define and execute" approach.

Witshel et al. [16] identified the need to make business process execution more flexible. They start from the insight that while offering valuable context information traditional business process modeling approaches are too rigid to capture the actual way processes are executed. This is also the case in software engineering, where the execution is even less predictable. The paper explores how practitioner knowledge can be leveraged to enhance BPMN [17] process models, by allowing workers to leave semi-structured comments on BPMN activities. Our contribution, while focused on software processes (instead of business processes) shares this feedback approach, where the process model is enriched with information extracted from actual execution.

In [18], Grudin et al. proposed a methodology for the design of collaborative environment which is halfway between the top-down approach of put-all-the-knowledge-in-at-the-beginning and the bottom up approach of just-provide-an-empty-framework. Witshel et al. [16] used this approach to design a "task pattern" formalism (task patterns are defined as "*abstractions of tasks that provide information and experience that is generally relevant for the task execution. By abstraction we mean common features of a family of similar tasks, which aim at the same goals under similar conditions*"). Task patterns are instantiated; by assigning real persons to the positions defined in the task pattern (concerned people can refuse or accept). A user can enhance a task pattern while executing it. The enhancements are stored locally, reused automatically for subsequent instances of the task pattern, and can be published so others can use it.

Van Der Aalst et al. [19] have also addressed the issue of taking into account instances of role, task, and product. They noted that the number of these instances cannot always be known in advance, and they should therefore be given distinct identities. However, their language, YAWL, is a workflow language, whose main goal is to contribute to the modeling of general work patterns, by extending existing Petri net-based approaches. In the same vein, Little-JIL [20] is a graphical coordination description language, with the ability to integrate with separate systems for resource, artifact, and agenda management. Little-JIL also supports executions agents (which may be identified humans like in CM_SPEM).

II. CONCLUSION

This work introduced a meta-process for modeling and enacting collaborative processes, and showed how it can be applied with CM_SPEM, a SPEM extension for the description of collaboration. Our guiding principle is that collaboration should be conceptualized using the ideas most familiar to people collaborating. This is reflected in our choice of concepts and the way they are combined. Our meta-model has been validated by the meta-process which shows how to get from modeling with CM_SPEM to project plans.

We implemented - based on MDE techniques - an editor for CM_SPEM, and of a generator which is capable of generating project plans (for popular project planning tools of the market) from CM_SPEM process models.

We found current project management tools lacking in their support of collaboration, which results in loss of information when generating project plans. We plan to explore how these tools can be enhanced with the addition of relationships for example.

Future extensions include visualizations to help practitioners better understand CM_SPEM models, define a complete query API (Application Programming Interface) for retrieving information from CM_SPEM process models, and an ecosystem of tools which support collaboration using information from process models.

ACKNOWLEDGMENT

We thank the Galaxy partners for their contribution to this work.

REFERENCES

[1] Duncan Haughey, An Introduction to Microsoft Project

[2] http://www.ganttproject.biz/

[3] Marco Kuhrmann, Georg Kalus, Manuel Then, Eugen Wachtel, From Design to Tools: Process Modeling and Enactment with PDE and PET, Proceedings of Third International Workshop on Academic Software Development Tools and Techniques (WASDeTT-3), co-located with the 25th IEEE/ACM International Conference on Automated Software Engineer.

[4] M. Gnatz, M. Deubler and M. Meisinger, Towards an integration of process modeling and project planning, 5th International Workshop on Software Process Simulation and Modeling (ProSim 2004)

[5] Software & Systems Process Engineering Meta-Model Specification, 24/08/2008

[6] www.omg.org

[7] Object Constraint Language, OMG Available Specification, Version 2.0, May 2006

[8] Faye C. Budlong, Paul A. Szulewski, Ralph J. Ganska, Process Tailoring for SoftwareProject Plans, January 1996

[9] Marc Pantel. The TOPCASED project: a Toolkit in OPen source for Critical Applications &SystEms Design. Dans : Model-DrivenDevelopmentTool Implementers Forum (MDD-TIF 2007), Zurich, 24/06/2007, Domain Specific Modelling Forum (DSMF), juin 2007.

[10] http://msdn.microsoft.com/en-us/library/bb968652.aspx

[11] Kedji K.A, Coulette B, Nassar, M, Lbath R., Ton That M.T. : Collaborative Processes in the Real World: Embracing their Essential Nature. In proc. International Symposium on Model Driven Engineering: Software & Data Integration, Process Based Approaches and Tools - colocated with ECMFA 2011, Birmingham, 06/06/2011-07/06/2011, Springer, June 2011.

[12] Kedji K.A, Coulette B, Lbath R., Nassar, M : Modeling Ad-hoc Collaboration for Automated Process Support. In proc. Software Quality Days (SWQD) 2012, Vienna, 17/01/2012-19/01/2012, Springer, January 2012 (to appear).

[13] Polanyi, M., Sen, A.: The tacit dimension. University of Chicago press (2009)

[14] Killisperger, P., Stumptner, M., Peters, G., Grossmann, G., Stückl, T.: A Framework for the Flexible Instantiation of Large Scale Software Process Tailoring. New Modeling Concepts for Today's Software Processes (2010) 100–111

[15] Porres, I., Valiente, M.: Process definition and project tracking in model driven engineering. Product-Focused Software Process Improvement (2006) 127–141

[16] Witschel, H., Hu, B., Riss, U., Thönssen, B., Brun, R., Martin, A., Hinkelmann, K.: A Collaborative Approach to Maturing Process-Related Knowledge. Business Process Management (2010) 343–358

[17] OMG: Business process model and notation, version 1.2. http://www.omg.org/spec/BPMN/1.2/ (2009)

[18] Grudin, J., McCall, R., Ostwald, J., Shipman, F.: Seeding, Evolutionary Growth, and Reseeding: The Incremental Development of Collaborative Design Environments. Coordination theory and collaboration technology (2001) 447

[19] W. Van Der Aalst and A. Ter Hofstede. "YAWL : yet another workflow language".Information Systems, Vol. 30, No. 4, pp. 245–275, 2005

[20] A. Cass, A. Lerner, E. McCall, L. Osterweil, S. Sutton Jr, and A. Wise. "Little-JIL/Juliette : a process definition language and interpreter". In : Software Engineering, 2000. Proceedings of the 2000 International Conference on, pp. 754–757,IEEE, 2000.

[21] M.T. Ton That, H. N. Tran, B. Coulette. Developement of MDE tools for supporting collaborative processes. Master Thesis, University of Toulouse, June 2011.